Hello World Walkthrough

Full source (https://github.com/mozilla/pdf.js/blob/master/examples/learning/helloworld.html)

PDF.js heavily relies on the use of Promises (https://developer.mozilla.org/docs/Web/JavaScript/Reference/Global_Objects/Promise). If promises are new to you, it's recommended you become familiar with them before continuing on.

This tutorial shows how PDF.js can be used as a library in a web browser. examples/ (https://github.com/mozilla/pdf.js/tree/master/examples) provides more examples, including usage in Node.js (at examples/node/ (https://github.com/mozilla/pdf.js/tree/master/examples/node)).

Document

The object structure of PDF.js loosely follows the structure of an actual PDF. At the top level there is a document object. From the document, more information and individual pages can be fetched. To get the document:

```
pdfjsLib.getDocument('helloworld.pdf')
```

Remember though that PDF.js uses promises, and the above will return a `PDFDocumentLoadingTask` instance that has a `promise` property which is resolved with the document object.

```
var loadingTask = pdfjsLib.getDocument('helloworld.pdf');
loadingTask.promise.then(function(pdf) {
  // you can now use *pdf* here
});
```

Page

Now that we have the document, we can get a page. Again, this uses promises.

```
pdf.getPage(1).then(function(page) {
  // you can now use *page* here
});
```

Rendering the Page

Each PDF page has its own viewport which defines the size in pixels(72DPI) and initial rotation. By default the viewport is scaled to the original size of the PDF, but this can be changed by modifying the viewport. When the viewport is created, an initial transformation matrix will also be created that takes into account the desired scale, rotation, and it transforms the coordinate system (the 0,0 point in PDF documents the bottom-left whereas canvas 0,0 is top-left).

```
var scale = 1.5;
var viewport = page.getViewport({ scale: scale, });

var canvas = document.getElementById('the-canvas');
var context = canvas.getContext('2d');
canvas.height = viewport.height;
canvas.width = viewport.width;

var renderContext = {
  canvasContext: context,
  viewport: viewport
};
page.render(renderContext);
```

Alternatively, if you want the canvas to render to a certain pixel size you could do the following:

```
var desiredWidth = 100;
var viewport = page.getViewport({ scale: 1, });
var scale = desiredWidth / viewport.width;
var scaledViewport = page.getViewport({ scale: scale, });
```

Interactive examples

Hello World with document load error handling

The example demonstrates how promises can be used to handle errors during loading. It also demonstrates how to wait until page loaded and rendered.

Hello World using base64 encoded PDF

The PDF.js can accept any decoded base64 data as an array.

Previous/Next example

The same canvas cannot be used to perform to draw two pages at the same time – the example demonstrates how to wait on previous operation to be complete.